

Software Reference Manual
for the
SEGA Mark III Console

Table of Contents

The Sega Mark III Game Console	3
The CPU	4
Reset	4
Interrupt System	4
NMI	4
INT	5
The Video Display Processor (VDP)	6
Video RAM Organization	6
Color	6
Background System	7
Scroll Inhibit Bits	7
Sprites	8
Sprite Attribute Table	8
Sprite Attribute Table Organization	9
Sprite/Background Display Priority	10
Color RAM	10
Character Patterns	10
An Example Character Pattern	11
VDP Registers	12
Read VDP RAM	12
Write VDP RAM	12
Write VDP Register	13
Write VDP Color RAM	13
Color RAM Addresses	14
VDP Register Updates	14
VDP Register Description	15
Register 0 - VDP Control	15
Register 1 - VDP Control	15
Register 2 - Screen Map Base Address	16
Register 3	16
Register 4	16
Register 5 - Sprite Attribute Table (SAT) Base Address	16
Register 6 - Sprite Pattern Base Address	17
Register 7 - Border Color	17
Register 8 - Horizontal Scroll	17
Register 9 - Vertical Scroll	17
Register 10 - Raster Line Interrupt	18
The Memory Management System	19
System RAM	19
Memory Enables	19
Memory Selection at Power-On	20
Memory Maps	21
32 Kilobytes	21
128 Kilobytes	21
The Cartridge Header	22

The Input/Output System	23
Input Ports Bit Assignments	24
How the Gun Works	25
How the Trackball Works	26
The Programmable Sound Generator (PSG)	27
Tone Generator Frequency	27
Noise Generator Control	28
Attenuators	29
Appendix A - VDP Registers	30
Appendix B - Input/Output Ports	31
Appendix C - Sample Code to Read Gun	32
Appendix D - Sample Code to Read TrackBall	37
Developer's Notes	39
280 Memory Map	39
Memory Control Registers	39
4M Bit ROM	40
280 Address \$0000 - \$03FF	40
Supporting Illustrations	41
Screen Map	41
Early Clock Bit	41
Background System	42
Sprite System	43
Memory Maps	44

The Sega Mark III Game Console

This manual describes the SEGA Mark III game console hardware. The manual is divided into five sections:

- The CPU
- The Video Display Processor (VDP)
- The Memory Management System
- The Input/Output System
- The Programmable Sound Generator (PSG)

Each section begins with a general description, followed by a detailed description of every control bit. Several pictorial diagrams at the back of the manual tie together all of the concepts described in the manual. If you wish to see how a particular feature fits into the total system, refer to these back pages.

HEADINGS

Important topics inside each section is marked as above with a "heading". This allows quick scanning for specific items of interest.

FORWARD REFERENCES

In some sections it is impossible to describe the system without making references to technical details described further forward in the manual. For example you will find reference to "Sprite Interrupts" in the CPU section, before the notion of a sprite is discussed in the VDP section.

For this reason you might find it helpful to browse the first parts of each section to get a feel for the total system before diving into the detailed descriptions.

The system will hereafter be referred to as the "Mk3"

THE CPU

The Mk3 uses a Z80A microprocessor, clocked at 3.58 MHz.

The implementation of Z80 features as they pertain to the Mk3 are described in this section.

RESET

The Z80A executes a RESET cycle when the base unit is turned on. This is the only Z80A reset mechanism implemented.

The momentary RESET button on the console does not control Z80A reset; it is attached to an input port for software polling.

INTERRUPT SYSTEM

The Mk3 implements two of the Z80 interrupts, the Non-maskable Interrupt (NMI), and the Maskable Interrupt (INT).

NMI

The NMI pin is connected to the console "PAUSE" button. Pressing this button causes the Z80 to execute a Restart instruction at location \$66. This acts as a simple subroutine call: the PC is pushed onto the stack and a jump to \$0066 is performed.

This interrupt is "edge-triggered", which means that if the PAUSE button is held down, you will get a single interrupt only.

NOTE: The last instruction of your NMI routine must be "RETN", Return from Non-Maskable Interrupt.

The PAUSE button is for the exclusive use of your program; you must set up the transfer of control at \$0066 to your PAUSE routine.

The most common PAUSE routine will toggle a software "pause" flag and turn off the sounds if this flag has just been set. The INT routine, which is periodically activated, then checks the condition of the pause flag and does no processing if it is set (it loops on the pause flag).

This method allows the pause feature to be implemented without turning the interrupts off and on (with a DI and later an EI instruction). This DI-EI combination can cause a spurious interrupt in the Z80.

The NMI cannot be inhibited.

INT

This interrupt is enabled with an "EI" instruction, and disabled with a "DI" instruction.

The Mk3 hardware supports the "mode 1" interrupt only. The boot

ROM executes the "IM 1" instruction at power-on.

In mode 1, activating the Z80A INT pin causes a Restart instruction to be executed at location \$38 if interrupts are enabled.

This interrupt can be activated by two events in the VDP chip:

1. Vertical Blanking Interval
2. Horizontal Line Counter

Here is a skeleton program for an Interrupt response routine:

```
(ac $0038)
    JP    INT

INT:
    PUSH  AF
    IN    A, [$0038]

    POP  AF
    EI
    RET
```

The first step is to save the working registers and flags. Other "PUSH" instructions should be added for any registers your interrupt routine will use.

Reading the I/O port at \$0038 does two things. First, it clears the interrupt request line from the VDP chip. Second, it provides VDP information as follows:

```
bit 7 --- 1: VBLANK Interrupt, 0: H-Line Interrupt (if enabled).
bit 6 --- 1: 9 sprites on a raster line.
bit 5 --- 1: Sprite Collision.
bits 4-0 (no significance).
```

The VDP VBLANK and H-Line interrupts are enabled with the IE0 and IE1 bits in VDP registers 0 and 1, as shown in the following chart:

Interrupt Enable Bits

IE0 (IE bit 5)	IE1 (IE bit 4)	Interrupt Source
0	1	H-Line Interrupt only.
1	1	Both H-Line and VBLANK.

When an INT is accepted, the interrupt system is turned off, exactly as if a "DI" instruction had been executed. Thus, the code to exit from an interrupt routine should restore the registers, execute an "EI" instruction, and return.

THE VIDEO DISPLAY PROCESSOR (VDP)

The Mk3 uses an advanced video controller chip called the VDP (Video Display Processor). The VDP is a Sega designed custom chip whose architecture resembles an enhanced TMS9918A.

The screen resolution of the VDP is 256 pixels horizontal, and 192 pixels vertical. A pixel can be shown as one of sixteen colors, selected from a palette of 64 different colors.

16 Kilobytes of RAM are dedicated to the video system. This RAM (called the video RAM) is attached directly to the VDP chip, and does not appear in the Z80 memory space. The Z80 reads and writes the Video RAM through VDP registers.

The VDP implements two independent graphic systems, a background system and a sprite system.

VIDEO RAM ORGANIZATION

The 16 Kilobyte Video RAM is divided into three sections:

1. A 1792 byte screen map. This map determines the placement of character cells (sometimes known as "tiles") on the background screen's 32 by 24 grid.
2. A 256 byte sprite attribute table. This table sets the X-Y coordinates and character number of up to 64 moveable objects, or sprites.
3. A 14,336 byte character generator. These are 8 by 8 pixel character patterns for the background, and/or 8(h) by 8(v) or 8(h) by 16(v) pixel character patterns for the sprites.

32 bytes define a single 8 by 8 pixel character. The character generator portion of the Video RAM thus is capable of defining up to 448 characters.

The placement of the three Video RAM sections is controlled by registers in the VDP. As we'll see, you have some choice over where they appear in the 16K Video RAM memory map.

COLOR

The attribute of color is carried in the character patterns in the character generator portion of the Video RAM. Each pixel in the character pattern contains 4 bits of information. A choice of sixteen colors is thus available for every pixel.

The sixteen available colors are not fixed. They are stored in a Color RAM inside the VDP chip. The Color RAM is organized as 32 6-bit values. The six bits provide four levels (2 bits) of red, green, and blue.

Note that 32 color values represent twice the addressing capability of the 4-bit pixels. Other VDP control bits allow selection of the 16 value color table from either the first half or the second half of the Color RAM.

BACKGROUND SYSTEM

The background system is composed of "characters" which are eight pixels high and eight pixels wide. The screen is organized as 768 visible characters, 32 horizontal by 24 vertical.

An additional 4 character rows exist below the 24 visible vertical rows. These additional rows are useful for scrolling new background information upwards onto the screen.

2048 bytes of the video RAM function as the screen map. This map defines the screen positions of 896 characters (768 visible).

At every character position, there is a 16-bit (2 byte) word which describes the following:

1. Which of 512 characters to display at the character position (9 bits).
2. Whether or not to flip the character horizontally or vertically (2 bits).
3. Which of two sets of 16 colors to use for the character (1 bit).
4. Whether sprites obscure the background, or vice versa (1 bit).
5. Three uncommitted bits, useful for software flags (3 bits).

The visible screen map occupies 1536 bytes of video RAM (768 characters, two bytes per character).

The nonvisible four rows of characters below the visible screen occupy 256 bytes (128 characters, two bytes per character).

This leaves 256 bytes of unused memory at the highest address of the 2048 byte screen map RAM. These 256 bytes are normally used to store the Sprite Attribute Table.

This memory could also be used to store eight character patterns, if the Sprite Attribute Table is located elsewhere. If the screen map is placed at \$3800 (the usual case), the character numbers for the unused 256 bytes of screen map RAM are \$1F0 through \$1F7.

SCROLL INHIBIT BITS

The background screen can be scrolled in one pixel increments both horizontally and vertically. Two "scroll inhibit" bits allow you to inhibit scrolling in two screen areas.

If HSI (HORIZ Scroll Inhibit) is set to 1, the 2 character high horizontal strip at the top of the screen does not scroll.

If VSI (VERT. Scroll Inhibit) is set to 1, the vertical band of 8 characters at the right edge of the screen does not scroll.

This feature simplifies the placement of score information at the top or right side of the screen. When the background screen playfield scrolls, the scores remain in place if the scroll inhibit bits have been set.

SPRITES

A sprite is an easily movable object. The VDP provides 64 independent sprites. A control bit controls the size of all sprites: 0 for an 8H by 8V sprite, 1 for an 8H by 16V sprite.

Each sprite uses a three byte entry in a Sprite Attribute Table (in Video RAM) to set it's vertical and horizontal position, and to select one of 256 characters to be displayed for the sprite.

A VDP control bit (Sprite Shift, bit 3 of R0) shifts all sprites 8 pixels to the left. This allows sprites to smoothly scroll off the left edge of the screen.

Sprite and background characters are drawn from the same "pool" of character patterns in video RAM. Thus sprites have the same color capabilities as background characters: 16 colors at a time from a choice of 64 colors.

Sprite colors are always taken from the second group of 16 colors in the color RAM.

Sprites do not scroll when the background scene is scrolled.

Sprites can be placed over or under other sprites. Sprites can also appear over or under characters in the background scene.

Up to eight sprites can occupy a single horizontal raster line. A VDP status register bit is provided to alert you when nine or more sprites are positioned on the same line. A horizontal raster line containing nine or more sprites is not displayed properly.

Another VDP status register bit indicates that two sprite patterns touched (collided).

SPRITE ATTRIBUTE TABLE

A 256 byte section of Video RAM functions as the Sprite Attribute Table. VDP register R5 is usually set to \$FF to position the Sprite Attribute Table at \$3F00, the last 256 bytes of Video RAM.

The Sprite Attribute Table is organized as shown on the table on the next page.

Sprite Attribute Table Organization

Address	Attribute
3F00	xpos #0
3F01	xpos #1
3F02	xpos #2
3F03	xpos #3
3F04	xpos #4
3F05	xpos #5
3F06	xpos #6
3F07	\$D0 (terminator)
.	.
.	.
.	.
3F1E	.
3F3F	xpos #63
3F40	64 unused bytes. (Can put two 32 byte character patterns here and address as char's \$1FA and \$1FB).
.	.
3F7F	.
3F80	xpos #0
3F81	char code #0
3F82	xpos #1
3F83	char code #1
3F84	xpos #2
3F85	char code #2
3F86	.
3F87	.
.	.
3FEE	xpos #63
3FFF	char code #63

The special code \$D0 is placed in a vertical position code to tell the sprite hardware to stop searching the list for sprites. All sprites which occur after the \$D0 entry are disabled.

When two sprites overlap, the higher numbered sprite will be displayed on top. The overlap priority is thus set by the positions of the sprites in the table.

The horizontal position bytes locate the upper left corner of the sprite at one of 255 horizontal coordinates on the screen.

An xpos=0 puts the sprite on the left column of the screen (left 8 pixels). An xpos=255 puts the sprite at the last pixel column; only the left column pixels of the sprite are seen and the other 7 columns in the sprite pattern are blanked.

This makes it easy to smoothly scroll a sprite off the **right** edge of the screen.

Setting a VDP Sprite Shift bit (80, bit 3) shifts all sprite patterns eight pixels to the left. This allows sprites to be smoothly scrolled off the **left** edge of the screen.

If smooth appearing and disappearing of sprites is desired on **both** edges of the screen, another VDP control bit (80, bit 5) can be set to 1 to blank the left column of characters.

Thus if 80 bit 3 is set to 0 (no shift left), and 80 bit 5 is set to 1 (blank left column), the screen is reduced to 31 character columns, and sprites enter and leave the screen smoothly. The left edge is handled by the fact that sprite xpos=0 puts the sprite in the left column, which is blanked in this mode.

SPRITE/BACKGROUND DISPLAY PRIORITY

Sprites can appear in front of or behind the background scene. This is controlled by bit 12 of the 16-bit background character code. If this bit is set to 0, all sprites appear over the background. If this bit is set to 1, background colors #1-15 appear over the sprites. Background color #0 always appears under the sprites.

COLOR RAM

The VDP chip contains a color RAM of 32 6-bit values. These values are formatted as "write only" 8-bit bytes:



where R1-R0, G1-G0 and B1-B0 define four intensities for red, green, and blue as follows:

R, G, B-1	R, G, B-0	Intensity
0	0	0/3
0	1	1/3
1	0	2/3
1	1	3/3 (brightest)

The color RAM is organized as two banks of 16 colors each. Colors are selected from either bank with a four bit color code.

CHARACTER PATTERNS

The actual dot patterns for screen images are called characters. Character patterns are used by both the background system and the sprites.

Every character is made up of 32 bytes. These bytes are organized as eight 4 byte groups, where each 4 byte group forms one row of pixels.

Four bits are required to select 16 colors. These bits come from corresponding bits in the four planes. The table on the next page shows a 32 byte character pattern. "0's" are shown as periods for clarity.

An Example Character Pattern

Byte	b7	b6	b5	b4	b3	b2	b1	b0	
0	1	1	1	1	1	1	1	1	e0 FIRST PIXEL ROW: '0101' = color #5
1	.	.	.	.	.	.	.	.	e1
2	1	1	1	1	1	1	1	1	e2
3	.	.	.	.	.	.	.	.	e3
4	.	.	.	.	.	.	.	.	e0 SECOND PIXEL ROW: '1101' = color #12,
5	.	.	.	.	.	.	.	.	e1 '0001' = color #1.
6	.	.	.	.	.	.	.	.	e2
7	.	.	.	.	.	.	.	.	e3
8	.	.	.	.	.	.	.	.	e0 THIRD PIXEL ROW
9	.	.	.	.	.	.	.	.	e1
10	1	1	1	1	1	1	1	1	e2
11	1	1	1	1	1	1	1	1	e3
12	.	.	.	.	.	.	.	.	e0 FOURTH PIXEL ROW
13	.	.	.	.	.	.	.	.	e1
14	1	1	1	1	1	1	1	1	e2
15	1	1	1	1	1	1	1	1	e3
16	.	.	.	.	.	.	.	.	e0 FIFTH PIXEL ROW
17	.	.	.	.	.	.	.	.	e1
18	.	.	.	.	.	.	.	.	e2
19	.	.	.	.	.	.	.	.	e3
20	.	.	.	.	.	.	.	.	e0 SIXTH PIXEL ROW
21	.	.	.	.	.	.	.	.	e1
22	.	.	.	.	.	.	.	.	e2
23	.	.	.	.	.	.	.	.	e3
24	.	.	.	.	.	.	.	.	e0 SEVENTH PIXEL ROW
25	.	.	.	.	.	.	.	.	e1
26	.	.	.	.	.	.	.	.	e2
27	.	.	.	.	.	.	.	.	e3
28	.	.	.	.	.	.	.	.	e0 EIGHTH PIXEL ROW '0011' = color #2
29	.	.	.	.	.	.	.	.	e1
30	.	.	.	.	.	.	.	.	e2
31	.	.	.	.	.	.	.	.	e3

This pattern represents a multicolor letter "C".

The top section is color #5 (reading corresponding bits from MSB to LSB, 0101).

The vertical section is color #12 (1100).

The bottom section is color #2 (0010).

Which colors these represent depends on the values stored in color RAM locations 2, 5 and 12.

NOTE: the 8 by 8 character background color is #0 (0000).

VDP REGISTERS

The VDP is controlled by eleven internal 8-bit registers. This section describes the method by which the Z80A accesses these registers, and then discusses the function of the individual registers.

The Z80 "sees" the VDP chip through two I/O port locations, \$BE and \$BF. I/O location \$BF is the COMMAND register for a write, and the STATUS register for a read.

I/O port \$BE is the read/write DATA register.

The Command Register is written twice in succession for all command operations, the data register may be read or written any number of times in succession, depending on the operation.

Because the command register is sequence sensitive, requiring a pair of out bytes per operation, a DI instruction should precede command register updates. This prevents, for example, an interrupt service routine that reads the VDP status register from disrupting the two byte synchronization.

There is a timing constraint for accessing the VDP chip.

The VDP chip cannot process data any faster than the following rates:

- 16 Z80A T-States during VBLANK
- 29 Z80A T-States during active video

This means that you should never issue two consecutive OUT or IN instructions to the VDP, they should be separated by at least a NOP instruction.

This constraint applies to Video RAM and Color RAM as well as the internal registers.

A two bit mode field in bits 7 and 6 of the second COMMAND byte determines one of four operations:

b7	b6	Operation
0	0	Read 16 Kbyte VDP RAM
0	1	Write 16 Kbyte VDP RAM
1	0	Write VDP Register
1	1	Write Color RAM

To set up for the first two operations, read/write Video RAM the two COMMAND bytes have the following format:

Read VDP RAM

First byte written to \$BF	----->
	A7 A6 A5 A4 A3 A2 A1 A0
Second byte written to \$BF	----->
	0 0 A13 A12 A11 A10 A9 A8

Write VDP RAM

First byte written to \$BF	----->
	A7 A6 A5 A4 A3 A2 A1 A0
Second byte written to \$BF	----->
	0 1 A13 A12 A11 A10 A9 A8

After the COMMAND bytes have been output, data can be read at Input Port \$BE or written at Output Port \$BE.

The VDP has an address auto-increment feature. Once the Video RAM starting address has been loaded, the data (I/O port \$BE) may be repeatedly accessed, and the address will automatically increase by one byte for every access.

NOTE: The autoincrement feature works only with all reads or all writes. Changing from read to write or vice versa requires two writes to the command register to reselect the mode.

The COMMAND register format for writing a VDP register (they are write-only) is shown below:

Write VDP Register

First byte written to \$BE	d7 d6 d5 d4 d3 d2 d1 d0
Second byte written to \$BE	1 0 0 0 x3 x2 x1 x0

The register number is r3-r0, and ranges from 0 through 10 (\$00 through \$0A). The data to be written to the register is d7-d0.

For the "write register" operation only, there is no write to the DATA register at \$BE, since the data (d7-d0) is contained in the first byte of the COMMAND byte pair.

The final command mode allows writing color values to the write-only VDP Color RAM.

Write VDP Color RAM

First byte written to \$BE	0 0 0 0 A4 A3 A2 A1 A0
Second byte written to \$BE	1 1 0 0 0 0 0 0

After loading this pair of bytes to the COMMAND register, the color RAM data is written to Output Port \$BE in the following format:

Color RAM Data written to \$BE	b0 b1 b2 b3 g0 g1 g2 g3 r0 r1 r2 r3
-----------------------------------	---

Where b1-b0 set four intensities for blue, g1-g0 set four intensities for green, and r1-r0 set four intensities for red.

As with the Video RAM, repeated writes to Output Port \$BE automatically increment the Color RAM address. Color RAM addresses are mapped as shown on the next page.

Color RAM Address	Color Number	
0000 (\$00)	Bank1, Color 0	(Note: Background colors can be taken from either the first group of sixteen or the second group of sixteen colors)
0001 (\$01)	Bank1, Color 1	
0002 (\$02)	Bank1, Color 2	
0003 (\$03)	Bank1, Color 3	
0004 (\$04)	Bank1, Color 4	
0005 (\$05)	Bank1, Color 5	
0006 (\$06)	Bank1, Color 6	
0007 (\$07)	Bank1, Color 7	
0008 (\$08)	Bank1, Color 8	
0009 (\$09)	Bank1, Color 9	
000A (\$0A)	Bank1, Color 10	
000B (\$0B)	Bank1, Color 11	
000C (\$0C)	Bank1, Color 12	
000D (\$0D)	Bank1, Color 13	
000E (\$0E)	Bank1, Color 14	
000F (\$0F)	Bank1, Color 15	
1000 (\$10)	Bank2, Color 0	(Note: Border colors and sprite colors are taken from this second group of sixteen colors)
1001 (\$11)	Bank2, Color 1	
1002 (\$12)	Bank2, Color 2	
1003 (\$13)	Bank2, Color 3	
1004 (\$14)	Bank2, Color 4	
1005 (\$15)	Bank2, Color 5	
1006 (\$16)	Bank2, Color 6	
1007 (\$17)	Bank2, Color 7	
1008 (\$18)	Bank2, Color 8	
1009 (\$19)	Bank2, Color 9	
100A (\$1A)	Bank2, Color 10	
100B (\$1B)	Bank2, Color 11	
100C (\$1C)	Bank2, Color 12	
100D (\$1D)	Bank2, Color 13	
100E (\$1E)	Bank2, Color 14	
100F (\$1F)	Bank2, Color 15	

The command register has one additional function. When it is read with an IN (SRF) instruction, it functions as an interrupt status register, and also clears interrupt requests from the VDP chip.

VDP REGISTER UPDATES

In general, all VDP updates should be done while the TV's video signal is blanked. If operations that affect the screen appearance are not done during screen blanking, the screen will appear to have "noise" in the picture.

Your program can detect horizontal and vertical blanking intervals using the interrupt system and VDP flag bits. This capability is discussed in the CPU section.

It is not necessary to synchronize VDP initialization with blanking intervals, since the screen can be blanked prior to the setup operations, and then turned on when the initialization is complete.

VDP REGISTER DESCRIPTION

Register 0

Register 1

R0 and R1 set the VDP display and interrupt mode

R0: b7	VSE	Vertical Scroll Inhibit (right 8 char columns)
		0: Scroll along with background.
		1: Fixed (no scroll).
b6	HEI	Horizontal Scroll Inhibit (top 3 char rows).
		0: Scroll along with background.
		1: Fixed (no scroll).
b5	LCR	Left Column Blank.
		0: Normal display.
		1: Blank left char column (border color).
b4	SEI	Interrupt Enable.
b3	SS	Sprites Shift.
		0: Normal sprite position.
		1: Shift all sprites 8 pixels to the left.
b2	1	(Always set this to 1).
b1	1	(Always set this to 1).
b0	ES	External Sync.
		(Always set this to 0).
R1: b7	1	(Always set this to 1).
b6	DIS	Display ON.
		0: Blank whole screen.
		1: Normal video.
b5	SEI	Interrupt Enable.
b4	0	(Always set this to 0).
b3	0	(Always set this to 0).
b2	0	(Always set this to 0).
b1	SS	Sprites Size.
		0: All sprites are 8 by 8.
		1: All sprites are 8 by 16.
b0	0	(Always set this to 0).

The two bit interrupt control field is defined as follows:

IE	IEI	Enabled Interrupt:
0	0	Master Lame.
1	1	VBLANK and Master Lame.

Register 2

R2 sets the base address for the screen map. It positions the 2048 screen map to one of eight starting addresses, on 2048 byte boundaries, according to the following table:

R2 Value	Screen Map Base Address
\$FF	\$3800 ← Normal Setting
\$FD	\$3000
\$FB	\$2800
\$F9	\$2000
\$F7	\$1800
\$F5	\$1000
\$F3	\$0800
\$F1	\$0000

The normal setting for R2 is \$FF, which positions the base address at \$3800, the highest-addressed 2 KByte section of the 16 KByte Video RAM.

Register 3

R3 should always be set to \$FF.

Register 4

R4 should always be set to \$FF.

Register 5

R5 controls the base address for the Sprite Attribute Table. This 256-byte table can be positioned at one of 64 starting addresses in Video RAM.

The base address is formatted in R5 as follows:



The normal setting for R5 is \$FF, which positions the Sprite Attribute Table at \$3F00. When R2 is \$FF, placing the screen map at \$3800, and R5 is \$FF, the unused top 256 bytes of screen map memory is occupied by the 256-byte Sprite Attribute Table.

Register 6

R6 sets the base address for the sprite patterns. Only two values are valid for R6:

R6 Value	Sprite Patterns
5FB	First 8K of Video RAM ← Normal Value
5FF	Second 8K of Video RAM

The preferred location for the sprite patterns is in the first 8K of video RAM, since the full 8192 bytes are available for patterns. In the second 8K, 2048 bytes are lost to the screen map and Sprite Attribute Table. Thus 256 sprite patterns can be stored in the first 8K, but only 192 sprite patterns can be stored in the second 8K.

Register 7

R7 sets the border color. The border color is taken from the second bank of colors in VDP Color RAM.



Register 8

R8 sets the horizontal scroll value for the background scene. A value of 500 produces no scrolling. A value of 501 moves the background scene one pixel to the left, and moves the leftmost column of pixels to the rightmost column (the scrolling "wraps around").

Higher values of R8 "rotate" the screen horizontally to a maximum of 255 pixels for a value of 5FF. Note that horizontal scrolling resembles a rotating vertical cylinder, with graphic information disappearing from the left side of the screen and reappearing on the right side of the screen.

Register 9

R9 sets the vertical scroll value for the background scene. A value of 500 produces no scrolling. A value of 501 moves the background scene one pixel upward. The vertical scrolling resembles a rotating horizontal cylinder, but the "wrap around" point is not at the screen edge, as with horizontal scrolling. Rather, it is at the 28th character row. (Remember that the screen is organized as 28 character rows, with the top 24 displayed on the screen).

Register 9 is latched in vertical blanking.

Register 10

R10 controls the Raster Line Interrupt

The TV beam sweeps out horizontal lines (called raster lines) from left to right, moving from the top to the bottom of the screen. The visible picture contains 192 of these raster lines.

At the end of each raster line there is a brief blanking interval (HBLANK) in which the TV beam is turned off as the beam retraces from the right side to the left side of the screen. This HBLANK interval last approximately 10 microseconds.

It is desirable to receive an interrupt just after a selected raster line has been displayed. For example, you might want to change a color value after the beam has swept out the top half of the screen. In this case you would want to know when the 95th raster line has completed.

R10 allows you to do this. If you load R10 with 94, a Raster Line Interrupt Request will be generated every time the 95th horizontal line finishes it's scan.

The value loaded into R10 should be one less than the raster line you wish to "trigger" the interrupt.

The Raster Line Interrupt continues to generate interrupt requests as the beam sweeps the screen. The table on the next page shows the settings of R10 and the raster lines that generate interrupt requests at HBLANK time:

R10 Value	Interrupt Requests at these HBLANK times
\$C0-\$FF	None
\$00	1, 2, 3, 4, 5, ..., 191
\$01	2, 4, 6, 8, 10, ..., 190
\$02	3, 6, 9, 12, ..., 189
\$03	4, 8, 12, 16, ..., 188
(\$C0)	(\$C0)

There are two special cases. \$FF turns off the interrupt requests, and \$00 gives a Raster Line interrupt request for every horizontal line.

There is a one interrupt delay between loading R10 and having the value take effect. For example, if you set R10 for line 20, and respond to the interrupt by resetting R10 to 150, the very next Raster Line Interrupt will occur at 20, and all subsequent ones will occur at 150.

The VDP registers are initialized at power on to the following values:

R0	\$16	mode
R1	\$A0	mode
R2	\$FF	Screen Map Table Base
R3	\$FF	(Always \$FF)
R4	\$FF	(Always \$FF)
R5	\$FF	Sprite Table Base
R6	\$FF	Sprite Pattern Table Base
R7	\$01	Border color #0
R8	\$01	Scroll-H
R9	\$01	Scroll-V
R10	\$FF	N-line Interrupt (\$FF=\$FF)

MEMORY MANAGEMENT

The Mk3 base unit contains 8 kilobytes of ROM and 8 Kilobytes of RAM. Game program ROM is plugged into the unit by using one of two slots.

The front slot accepts a slim cartridge that resembles a credit card. The present capacity of this card is 32 Kilobytes. A 128 Kilobyte card is in the works.

The top slot accepts a more conventional plastic-cased game cartridge. 128 Kilobyte cartridges are now in production. 256 Kilobyte and larger cartridges are in the works.

The Z80A processor is capable of addressing 64 kilobytes of memory. Because the added game cartridge memory can exceed 64 Kilobytes, a simple memory management system is implemented in the Mk3.

When the Mk3 is turned on, the memory map contains an 8 kilobyte ROM at \$0000, and an 8 Kilobyte RAM at \$C000.

SYSTEM RAM

The RAM is mapped into two places, \$C000-\$DFFF and \$E000-\$FFFF. This RAM is used for Z80A stack and scratchpad use.

The top eight locations of RAM are reserved for memory management control registers, and should not be used by a game program.

For future compatibility, the RAM should be addressed at \$C000-\$DFFF. The stack should be placed at \$DFF8.

MEMORY ENABLES

Output port \$3E controls the memory enables. Individual bits in this port enable the System ROM, System RAM, Card Slot, Cartridge Slot, and External Slot. (The external Slot is not presently used).

Output port \$3E has the following format:

OUT (\$3E):	

bit 0	don't care
bit 1	don't care
bit 2	0= enable joysticks 1= disable joysticks
bit 3	0= enable base unit OS ROM 1= disable base unit OS ROM
bit 4	0= enable internal OS RAM 1= disable internal OS RAM
bit 5	0= enable card ROM 1= disable card ROM
bit 6	0= enable cartridge ROM 1= disable cartridge ROM
bit 7	0= enable external port 1= disable external port

MEMORY SELECTION AT POWER-ON

When the system is turned on, the Z80A begins executing code at location 50000 in the resident 8 KiloByte ROM. This ROM code performs the following steps:

1. The system is initialized.
2. A small program is copied from ROM into the RAM (at 5C700).
3. The program jumps to 5C700 to execute this program.
4. All of the system ROM spaces are disabled. This is why the program is copied to RAM and run there--the ROM spaces can then be checked one by one.
5. The ROM spaces are individually enabled and checked for the presence of memory. They are checked in the following order:
 - A. Front card slot
 - B. Top cartridge slot
 - C. External (rear) slot.
6. If memory is detected in any of these three slots, the slot is enabled and the Z80A executes a jump to location 50000.
7. If no memory is detected, the console ROM is enabled and an "Insert a cartridge" message is put on the screen.

Notice that if both the card and cartridge are plugged in, the card will be enabled because it is checked first.

A game program is written as a completely self-contained Z80A application program with origin 50000. A game program must take care of all system "housekeeping", for example setting up the interrupt vectors.

Once a game is running, there is no access to the host ROM.

MEMORY MAPS

The figures on the following page show memory maps for two presently defined memory sizes: 32 Kilobytes, and 128 Kilobytes.

Figure M-1 shows the Z80A memory map at power-on.

32 KILOBYTES

Figure M-2 shows the Z80A memory map for the 32 Kilobyte card. This is the simplest case: 32 Kilobytes starting at location \$0000.

128 KILOBYTES

Figure M-3 shows the memory map for a 128 Kilobyte cartridge. (This is referred to in the Sega literature as a "Mega"(bit) cartridge).

As with the 32 KByte memory, there is a contiguous section of ROM in the bottom half of Z80A memory. This section is always available to the Z80A.

Additionally, there are six banks of 16 Kilobyte ROM at memory addresses \$8000-\$BFFF. Only one of these banks is available to the Z80A at a time. Whenever one is selected, the other five are inactive.

Which of the six banks is switched into the Z80A memory space is controlled by a register at memory location \$FFFF.

The Bank Control Register has the following format:

00000 0 0 0 0 0 b2 b1 b0

The banks are numbered from 2 through 7 (b2 b1 b0).

Although this write-only register exists in the memory cartridge, writes to the register are duplicated in the RAM at \$FFFF. This means that the bank select register appears to be read/write, even though what is actually read is the RAM image of the register, rather than the contents of the register itself.

THE CARTRIDGE HEADER

The presence of a cartridge is tested by checking for specific information in ROM. This information must be correct for the cartridge to be recognized.

Normally, Sega will add the correct information to the cartridge before production. The information below is given for reference only.

The header format is as follows (addresses shown for 32 KByte and larger cartridges)

At \$7FF0:

The string, "TMR SEGA " [\$4,4D,52,20,53,45,47,41,20,20]

At \$7FFA-\$7FFB

A 16-bit checksum for the ROM (low-to order). \$7FFA and \$7FFB are not included in this checksum.

At \$7FFC-\$7FFD

A 16-bit serial number, assigned by Sega.

At \$7FFE

A software revision number byte

At \$7FFF

ROM Size code

\$40	8K	Overder at: \$1FFF
\$48	16K	Overder at: \$3FFF
\$4C	32K	
\$4D	48K	
\$4E	64K	
\$4F	128K (1M)	
\$40	256K (2M)	

INPUT/OUTPUT (I/O) PORTS

Eight I/O ports are used by the MK3:

Port	Direction	Function
\$3E	out	Memory Enable
\$3F	out	Joystick port control
\$7E	in	Gun Spot (vertical)
\$7F	an out	Gun spot (horizontal) Programmable Sound Generator (PSG)
\$BE	i/o	VDP Data Register
\$BF	i/o	VDP Command/Status Register
\$DC	an	Joystick port
\$DD	an	Joystick port

Port \$3E is described in the MEMORY MANAGEMENT section.

Port \$7F is described in the PSG section.

Ports \$BE and \$BF are described in the VDP section.

The remaining ports deal with the attached controls, and are described in this section.

Ports \$DC-\$DD are connected to the two 9-pin connectors on the front of the game console. Although these ports are called "joystick ports", there are actually three devices that can be plugged into the front connectors:

1. The joystick controls which include a joystick and two push buttons.
2. The gun.
3. A "Sports Pad" that consists of a trackball and two push buttons.

INPUT PORTS BIT ASSIGNMENTS

The bits of input ports \$DC and \$DD are used for the joystick assembly, gun, and Sports Pad (Trackball). These two ports work in conjunction with port \$3F, which sets the direction of four of the bits, and provides two strobe bits for the Sports Pad. The following table shows how the three ports are interpreted for the joystick and gun options. All bits shown are inputs.

Port	Bit	Joystick	Gun
\$DC	0	F1-up	---
\$DC	1	F1-down	---
\$DC	2	F1-left	---
\$DC	3	F1-right	---
\$DC	4	F1-SHL	F1 Trigger
\$DC	5	F1-SH2	---
\$DC	6	F2-up	---
\$DC	7	F2-down	---
\$DD	0	F2-left	---
\$DD	1	F2-right	---
\$DD	2	F2-SHL	F2 Trigger
\$DD	3	F1-SH2	---
\$DD	4	RESET*	RESET*
\$DD	5	---	---
\$DD	6	---	F1 Light Pulse
\$DD	7	---	F2 Light Pulse
\$3F	0	1	1
\$3F	1	1	F1 Latch Enable
\$3F	2	1	1
\$3F	3	1	F2 Latch Enable
\$3F	4	1	1
\$3F	5	1	1
\$3F	6	1	1
\$3F	7	1	1

HOW THE GUN WORKS

The gun contains a lens and photocell which "sees" a spot on the TV screen. This spot is circular, and increases in diameter as the gun's distance from the screen increases.

When the TV beam reaches the area at which the gun is pointed, two things happen:

1. A negative pulse of approximately 20-30 microseconds is generated on port SDD bit 6 (Player 1) or port SDD bit 7 (Player 2). This pulse is wide enough to allow a software loop to recognize it. The width of this pulse will vary with the game setup and TV type.
2. The beam position is latched into horizontal and vertical registers, which are read at input ports \$7E (vertical) and \$7F (horizontal).

The beam position can be latched either from a gun plugged into the player 1 connector, a gun plugged into the player 2 connector, or both. [NOTE: The player 1 connector is the left connector when the game console is viewed from the front].

Two bits in output port \$3F determine which gun latches the beam position. These are shown as "P1 Latch Enable" and "P2 Latch Enable" in the preceding table.

The gun reading routine usually performs the following steps:

1. The trigger switch is checked. Normally the gun is not checked until someone pulls the trigger. (Data is continuously available from the gun, however)
2. When the trigger is pulled, wait for the next VBLANK.
3. At VBLANK, put up an all white screen (all color bytes are \$3F). This gives enough light intensity on the screen for the gun to read any position.
4. During the next scan, continuously check the light pulse bit for a high to low transition. (P1 Light Pulse is at input port SDD bit 6; P2 Light Pulse is at input port SDD bit 7).
5. When the pulse is detected, read the Horizontal Position Register at \$7F and the Vertical position Register at \$7E.
6. Wait for a low to high transition on the Light Pulse bit. Then repeat step 5 for all active horizontal lines.
7. When finished with one scan, restore the screen to the normal color values.

The gun responds to a circular spot on the TV, not a single pixel. As the circular spot is "seen" by the photocell in the gun, repeated negative pulses will occur on the Light Pulse bit for several consecutive horizontal scan lines.

The horizontal values of the detected circle will, as the vertical position register increments, decrease, increase, and then stop as the left edge of the scanned circle is read by the lens in the gun.

For this reason it is advisable to do some sort of averaging and extrapolation to arrive at a center position of the scanned circle.

HOW THE TRACKBALL WORKS

The interface to the trackball consists of six input bits and one output bit.

Trackball	dir	Player 1	Player 2
b0	in	SDC bit 0	SDC bit 6
b1	in	SDC bit 1	SDC bit 7
b2	in	SDC bit 2	SDC bit 0
b3	in	SDC bit 3	SDC bit 1
b4	in	SDC bit 4	SDC bit 2
b5	in	SDC bit 5	SDC bit 3
STROBE	out	SDC bit 5	SDC bit 7

Before using the trackball, the following two instructions should be executed to set the STROBE bit directions to output, and initialize the strobe signals to 1

NOTE: these instructions are executed in the power-on initialization

```
LD A,10100101b ; b5,b3 set strobe direction to output
OUT 103F81,A ; b5,b7 are actual strobe bit values
```

The strobe signal is then used to clock in four 4-bit nibbles b3-b0, which occur in the following order:

Nibble #	Data	When STROBE is,
	b3 b2 b1 b0	
1	X3 X2 X1 X0	LO
2	X3 X2 X1 X0	HI
3	Y3 Y2 Y1 Y0	LO
4	Y3 Y2 Y1 Y0	HI

The X,Y positions of the trackball are represented as X7-X0 and Y7-Y0. These are 8-bit counters which wrap around from 255 to 0.

As the ball is spun to the right, the X value increase; as it is spun to the left, they decrease. As the ball is spun upward, the Y values increase; as it is spun downward, they decrease.

The timing of the STROBE transitions is important. Here is the required sequence

Program Sequence to Read Trackball

```
(Start with STROBE HI).

1. STROBE LO.
2. Wait 80 microseconds.
3. Read first nibble.
4. STROBE HI.
5. Wait 40 microseconds.
6. Read second nibble.
7. STROBE LO.
8. Wait 40 microseconds.
9. Read third nibble.
10. STROBE HI.
11. Wait 40 microseconds.
12. Read fourth nibble.
13. Leave STROBE as HI.
```

Appendix D shows working code to read the trackball each interrupt.

PROGRAMMABLE SOUND GENERATOR (PSG)

The PSG contains four sound channels, consisting of three tone generators and a noise generator. Each of the four channels has an independent volume control (attenuator). The PSG is controlled through output port \$7F.

TONE GENERATOR FREQUENCY

The frequency (pitch) of a tone generator is set by a 10-bit value. This value is counted down until it reaches zero, at which time the tone output toggles and the 10-bit value is reloaded into the counter. Thus, higher 10-bit numbers produce lower frequencies.

To load a new frequency value into one of the tone generators, you write a pair of bytes to I/O locations \$7F according to the following format:



The R2:R1:R0 field selects the tone channel as follows:

R2	R1	R0	Tone Channel
0	0	0	#1
0	1	0	#2
1	0	0	#3

10-bit data is (msb) d9 d8 d7 d6 d5 d4 d3 d2 d1 d0 (lsb)

NOISE GENERATOR CONTROL

The noise generator uses three control bits to select the "character" of the noise sound. A bit called "FB" (Feedback) produces periodic noise or "white" noise.

FB	Noise Type
0	Periodic (like low-frequency tone)
1	White (Glas)

The frequency of the noise is selected by two bits NF1:NFO according to the following table:

NF1	NFO	Noise Generator Clock Source
0	0	Clock/2 (Higher pitch, "less coarse")
0	1	Clock/4
1	0	Clock/8 (Lower pitch, "more coarse")
1	1	Tone Generator #3

Note: "Clock" is fixed in frequency. It is a crystal controlled oscillator signal connected to the PSG.

When NF1:NFO is 11, Tone Generator #3 supplies the noise clock source. This allows the noise to be "swept" in frequency. This effect might be used for a jet engine rumup, for example.

To load these noise generator control bits, write the following byte to I/O port \$7F:

```
Out ($7F), ( 1 | 1 | 1 | 1 | 0 | 0 | FB | NF1 | NF0 | )
```

ATTENUATORS

Four attenuators adjust the volume of the three tone generators and the noise channel. Four bits A3:A2:A1:A0 control the attenuation as follows:

A3	A2	A1	A0	Attenuation
0	0	0	0	0 db (maximum volume)
0	0	0	1	2 db
0	0	1	0	4 db
0	0	1	1	6 db
0	1	0	0	8 db
0	1	0	1	10 db
0	1	1	0	12 db
0	1	1	1	14 db
1	0	0	0	16 db
1	0	0	1	18 db
1	0	1	0	20 db
1	0	1	1	22 db
1	1	0	0	24 db
1	1	0	1	26 db
1	1	1	0	28 db
1	1	1	1	-OFF-

NOTE: a higher attenuation results in a quieter sound.

The attenuators are set for the four channels by writing the following bytes to I/O location \$7F:

```

Tone Generator #1:  | 1 | 1 | 0 | 0 | 1 | A3 | A2 | A1 | A0 |
                    +-----+
Tone Generator #2:  | 1 | 1 | 1 | 1 | 1 | A3 | A2 | A1 | A0 |
                    +-----+
Tone Generator #3:  | 1 | 1 | 1 | 0 | 1 | A3 | A2 | A1 | A0 |
                    +-----+
Noise Generator:    | 1 | 1 | 1 | 1 | 1 | A3 | A2 | A1 | A0 |
                    +-----+

```

EXAMPLE

When the MK3 is powered on, the following code is executed:

```

LD    HL,CLATCH    ; clear table
LD    C,POD_PAT     ; pod pat is 0FF
LD    R,4           ; load four bytes
OTIR                ; write them
(etc.)

CLATCH  defb 0FF,00F,00F,0FF

```

This code turns the four sound channels off. It's a good idea to also execute this code when the PAUSE button is pressed, so that the sound does not stay on continuously for the pause interval.

Appendix A - VDP Registers

[illegible]

NOTE: First disable interrupts since VOP command register must be written to twice always (interrupts need the status register and VOP register addresses are sequence sensitive)

Appendix B - Input/Output Ports

000	SST		CAET		CAND		BAIN		GSDM		JOY		A		Z		HEDGE		SHARLES	

010	P2 TR10		P2 TR		P1 TR10		P1 TR		P2 TR10		P2 TR		P1 TR10		P1 TR		DATA DIRECTION		DATA DIRECTION	

DATA DIRECTION																				
AND OUTPUT DATA DIRECTION																				
I = Input, O = Output																				

020	V7		V6		V5		V4		V3		V2		V1		V0		---		Read	

Data Portion																				

030	H7		H6		H5		H4		H3		H2		H1		H0		---		Read	

Data Portion																				

040	-		-		-		-		-		-		-		-		Write		Programmable	

Based on Data																				
(P2) read...																				

050	-		-		-		-		-		-		-		-		Read		VOP Data	

Write: VOP Data																				

060	-		-		-		-		-		-		-		-		Write		VOP Command	

Write: VOP Command																				

070	INTER-		SPRATIC		SPRATIC		X		X		X		X		X		Read		VOP Status	

Data Portion																				

080	P2 TR		P1 TR		X		SECRET		P2 TR		P2 TR		P2 TR		P2 TR		---		Read	

Data Portion																				

090	P2 TR		P1 TR		X		SECRET		P2 TR		P2 TR		P2 TR		P2 TR		---		Read	

Data Portion																				

Appendix C - Sample Code to Read Gun

```

20
21
22
23      CDATA EQU      SCLOCK           ; Gun Status Work (1)
24      RVCHT EQU      CDATA+1         ; RAM Save Counter (1)
25      RVDATA EQU      RVCHT+1        ; V.M Counter Save Work (40H)
26      GCHMCT EQU      RVDATA+040H   ; (1)
27      GCHMHR EQU      GCHMCT+1      ; (1)
28      SMDOTF EQU      GCHMHR+10     ; Gun Shot Flag (1)
29      WPCST EQU      SMDOTF+1       ; W.Position Data (1)
30      WPCSE EQU      WPCST+1        ; V.Position Data (1)
31      SWDATA EQU      WPCSE+1        ; Switch Data (2)
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000

```

```

51:  ;-----
52:  ; ***** V & H DATA CHECK *****
53:  ;-----
54:  GUNCHK: LD A, (SHOOTF) ; Gun Shoot Flag
55:  OR A
56:  JZ RET
57:  RET
58:  OR A
59:  LD (SHOOTF),A
60:  GUNCHKI: LD A, (VCHNT) ; V & H Counter
61:  CP 3
62:  JZ C
63:  RET
64:  ;
65:  DEC A
66:  LD B,A
67:  LD HL,SHOOT+1
68:  LD R,SH
69:  GUNCHK: LD A, (HL) ; V,Counter
70:  INC R
71:  LD C,A
72:  INC HL
73:  INC HL
74:  INC HL
75:  OR A
76:  LD A, (HL)
77:  SUB C
78:  CP 3
79:  CALL NC,SHOOTL2
80:  DJNZ GUNCHK
81:  CALL GUNCHK2
82:  ;
83:  CALL CASH
84:  ;
85:  LD A, (HL) ; H-COUNTER READ
86:  CP CASH
87:  JZ NC
88:  AND A,A ; CARRY FLAG = CLEAR
89:  SUB 01EH ; A=A-1EH
90:  ;
91:  ;SHR-SH4M sRight ?? ????
92:  ;SHR-SH4M sLeft ?? ????
93:  ;
94:  SLA A ; A=A*2
95:  LD R,A
96:  REPT 3
97:  XRA
98:  ENEM
99:  AND A,SH ; CARRY FLAG = CLEAR
100:  ADD B ; A=A+A/16
101:  LD (SHOOT),A
102:  INC HL
103:  LD A, (HL) ; V,Counter Read
104:  AND A,A ; CARRY FLAG = CLEAR
105:  ;SUB 01EH ; A=A-1EH
106:  SUB 001H
107:  LD (VPSGT),A
108:  ; * Color Data Set *
109:  XRA
110:  OUT (SHFP),A
111:  LD A,SHCM
112:  OUT (SHFP),A
113:  LD BC,SHCONH
114:  LD HL,COLORTABL ; Color Data Table
115:  OTIR
116:  RET
117:  COLORTABL:
118:  DEFB 000H,000H,000H,000H,000H,000H,000H,000H
119:  DEFB 000H,000H,000H,000H,000H,000H,000H,000H
120:  DEFB 000H,000H,000H,000H,000H,000H,000H,000H
121:  DEFB 000H,000H,000H,000H,000H,000H,000H,000H
122:
123:
124:

```

```

1280
1281
1282: GCRRL2:
1283:     PUSH    HL
1284:     LD      HL, SCCHACT
1285:     LD      A, (HL)
1286:     CP      5
1287:     JP      NC, SCCHL1
1288:     SMC     (HL)
1289:     LD      HL, SCCHORR
1290:     PUSH    DE
1291:     LD      D, 0
1292:     LD      R, A
1293:     ADD     HL, DE
1294:     POP     DE
1295:     LD      HL, 0
1296:     LD      HL, 0
1297:     RET
1298:
1299: SCCHL1:
1300:     POP     HL
1301:     LD      HL, R, DE
1302:     RET
1303:
1304: SCCHL2:
1305:     LD      A, (HL)
1306:     JP      C, SCCHORR
1307:     LD      HL, SCCHACT
1308:     CP      1
1309:     JP      Z, SCCHORR
1310:     LD      R, A
1311:
1312: SCCHL3:
1313:     LD      A, (HL)
1314:     SMC     HL
1315:     CP      (HL)
1316:     JP      C, SCCHL3
1317:     SMC     SCCHL2
1318:     JP      SCCHL2
1319:
1320: SCCHL4:
1321:     SMC     C
1322:     SMC     SCCHL3
1323:
1324: SCCHL5:
1325:     JRCA
1326:     AND     07FH
1327:     LD      R, C
1328:     LD      C, A
1329:     LD      A, B
1330:     OR      A
1331:     JP      Z, SCCHL5
1332:     LD      HL, SCCHORR
1333:     XOR     A
1334:
1335: SCCHL6:
1336:     ADD     A, (HL)
1337:     SMC     SCCHL5
1338:     ADD     A, C
1339:
1340: SCCHL7:
1341:     ADD     A, A
1342:     LD      C, A
1343:     LD      B, 0
1344:     LD      HL, 0
1345:     ADD     HL, 0
1346:     RET
1347:
1348: SCCHL8:
1349:     LD      A, C
1350:     JP      SCCHL7
1351:
1352:
1353:
1354:
1355:
1356:

```



```

239: ;-----
240: ;
241: ;
242: ;
243: ;-----
244: ;
245: ;      1.0.0.0      ROM
246: ;      ----- CPG
247: ;      0.1.0.1      ROM
248: ;      1.1.0.0      old
249: ;      ----- AND
250: ;      0.1.0.0
251: ;      ----- CPG
252: ;      1.0.1.1
253: ;-----
254: ZSET:
255:     IN      A, [ROM]
256:     AND     0100
257:     LD      HL, SRDATA
258:     CPG
259:     LD      C, A
260:     ROR     [HL]
261:     LD      [HL], C
262:     INC     HL
263:     AND     C
264:     LD      [HL], A
265:     RET
266:
267:
268:
269:
270:
271:
272: ;-----
273: ;      *** Interrupt Jump Check ***
274: ;-----
275: JCRD:  SJMP  AF
276:         PUSH  AF
277:         LD    A, [CDATA]
278:         SEC   A
279:         JK    HL, INTR
280:         EX    [SP], HL
281:         POP   HL
282:         LD    HL, INTR0
283:         EX    [SP], HL
284:         PUSH  AF
285: INTR0:  POP   AF
286:         JP    INTR0

```

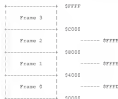
Appendix D - Sample Code to Read Track Ball

[illegible]

770					
780	0220h	03	OR	h	
790	0220h	57	LD	0, A	
800	0220h	38 87	LD	A, 00000111B	
810	0220h	03 3F	OUT	(R0WC), A	
820	0221h	06 8D	LD	R, 13	
830	0223h	10 FE	DNUI	5	
840	0229h	08 0C	IN	A, CP1_SMP(T)	
850	0229h	86 C6	AND	11000000h	
860	0229h	5F	LD	R, A	
870	022Ah	08 8D	IN	A, CP2_SMP(T)	
880	022Ch	86 83	AND	11h	
890	022Ch	83	OR	R	
900	022Fh	07	RLCA		
910	0240h	07	RLCA		
920	0241h	82	OR	0	
930	0242h	8D 44	RRD		
940	0244h	32 C01A	LD	(R2_TRIG), A	
950	0247h		TAACK_Y1:		
960	0247h	38 87	LD	A, 00000111B	
970	0249h	03 3F	OUT	(R0WC), A	
980	0249h	06 19	LD	R, 25	
990	024Dh	10 FE	DNUI	5	
1000	024Fh	08 0C	IN	A, CP1_SMP(T)	
1010	0251h	86 C6	AND	11000000h	
1020	0253h	0F	RRCA		
1030	0254h	0F	RRCA		
1040	0255h	8F	LD	R, A	
1050	0256h	08 8D	IN	A, CP2_SMP(T)	
1060	0256h	86 83	AND	11h	
1070	025Ah	0F	RRCA		
1080	025Ah	0F	RRCA		
1090	025Ch	83	OR	R	
1100	025Ch	57	LD	0, A	
1110	025Eh	38 87	LD	A, 00000111B	
1120	0260h	03 3F	OUT	(R0WC), A	
1130	0262h	06 8D	LD	R, 13	
1140	0264h	10 FE	DNUI	5	
1150	0266h	08 0C	IN	A, CP1_SMP(T)	
1160	0268h	86 C6	AND	11000000h	
1170	026Ah	5F	LD	R, A	
1180	026Bh	08 8D	IN	A, CP2_SMP(T)	
1190	026Dh	86 83	AND	11h	
1200	026Fh	83	OR	R	
1210	0270h	07	RLCA		
1220	0271h	07	RLCA		
1230	0272h	82	OR	0	
1240	0273h	8D 44	RRD		
1250	0275h	32 C01B	LD	(R2_TRIG), A	
1260	0278h	C9	RET		
1270					

Developer's Notes

111 溫哥華 - Memory Map



(2) Memory Control Registers

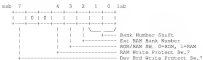
* 1FFFF - Frame 2 Control Register



* 18VFE - Frame 1 Control Register

* **IFVFO** - Frame 0 Control Register

* SEETC - Frame 2 NAM Control Register



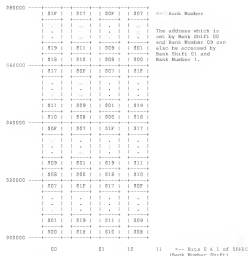
NOTE: When power is applied, the following data is set by the system ROM program:

FFFF = 2 FFFF = 3 FFFF = 0 FFFF = 6

Bit 7 of SPWPC is write protect bit for ROM Board (Just for development board only)

0 = N/A 1 = Read Only

(3) The 4K Bank ROM



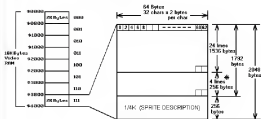
(4) ROM Address 00000 - 003FF

The lower 1K (1024 bytes) of the ROM address range is always mapped to the lower 1K (1024 bytes) of ROM to ensure that the reset and interrupt vectors were always available.

For example, if Bank #8 (ROM Address 320000 - 324000) were assigned to Frame 0 (RAM Address 50000 - 54000) then ROM Address 320000 - 3233FF cannot be accessed as ROM Address 00000 - 003FF remains mapped to ROM Address 000000 - 0003FF.

Supporting Illustrations

Screen Map



* If no vertical scroll is used, these character codes are available (use this memory to store 6 character patterns)



Early Clock Bit



